



# LCI Advanced Workshop 2025: Service Restoration & Install Repair

Alan Chapman  
HPC Systems Analyst  
Arizona State University Research Computing  
[alan.chapman@asu.edu](mailto:alan.chapman@asu.edu)

*This document is a result of work by volunteer LCI instructors and is licensed under CC BY-NC 4.0  
(<https://creativecommons.org/licenses/by-nc/4.0/>)*

# Agenda

- **Common Slurm Failures**  
Resource allocation, job submission, MPI issues
- **Core Infrastructure Failures**  
slurmctld, slurmdbd, slurmd daemon failures
- **Critical Recovery Procedures**  
Emergency restoration, database recovery
- **Monitoring & Prevention**  
Health checks, automated backups
- **Q&A & Discussion**

# Diagnostic Workflow Overview

## Standard Troubleshooting Sequence:

### *# User-level diagnostics*

<code>squeue -u \$USER -o "%i %t %r %S %M %l"</code>	<i># Job status</i>
<code>scontrol show job \$JOBID</code>	<i># Detailed job info</i>
<code>sacct -j \$JOBID --format=JobID,State,ExitCode</code>	<i># Accounting</i>
<code>tail -50 slurm-\$JOBID.out</code>	<i># Output logs</i>
<code>seff \$JOBID</code>	<i># Efficiency</i>

### *# System-level diagnostics (head node)*

<code>sinfo -N -l   grep -v idle</code>	<i># Problem nodes</i>
<code>systemctl status slurmctld slurmdbd</code>	<i># Service status</i>
<code>journalctl -u slurmctld -n 50</code>	<i># System logs</i>

Tip: Always start with `scontrol show job $JOBID` – it reveals 80% of issues

Image source: <https://slurm.schedmd.com/network.html>

# Resource Allocation Failures

## Node Unavailable Issues:

Error: "Requested node configuration is not available"

### # Diagnosis

```
sinfo -N -l | grep -v idle  
scontrol show node lci-compute-40-1
```

### # Resolution

```
scontrol update NodeName=lci-compute-40-1 State=resume Reason="cleared"  
sbatch -x lci-compute-40-1,lci-compute-40-2 script.sh # Exclude problem nodes
```

## QOS/Account Limits:

Error: "QOSMaxWallDurationPerJobLimit"

### # Check limits

```
sacctmgr show assoc user=$USER format=user,account,maxjobs,maxwall  
sprio -j $JOBID # Priority analysis
```

### # Fix

```
sbatch -t 23:59:00 -A different_account script.sh
```

# Hardware Detection Failures

## Memory Issues – OOM Kills

*# Error in slurmd.log: "Detected 1 oom-kill event(s)"*

### *# Diagnosis*

```
sacct -j $JOBID --format=JobID,MaxRSS,ReqMem,Elapsed  
dmesg | grep -i "killed process"
```

### *# Resolution*

```
sbatch --mem=64G script.sh          # Increase memory  
sbatch --mem-per-cpu=4G script.sh   # Per-CPU allocation
```

## GPU Problems

*# Error: "couldn't communicate with NVIDIA driver"*

### *# Diagnosis & Fix*

```
srun --gres=gpu:1 nvidia-smi  
sudo nvidia-smi -pm 1          # Enable persistence  
sudo systemctl restart slurmd  
scontrol update NodeName=node001 State=resume
```

# Controller (slurmctld) Failures

## Daemon Crash – Most Critical Failure

### *# Symptoms*

queue: Unable to contact slurm controller (connect failure)

### *# Diagnosis*

```
systemctl status slurmctld  
journalctl -u slurmctld -f  
tail -100 /var/log/slurm/slurmctld.log  
ss -tlnp | grep :6817
```

### *# Recovery*

```
systemctl start slurmctld
```

### *# Debug mode if fails:*

```
slurmctld -D -vvv
```

# Controller (slurmctld) Failures

## State File Corruption – Data Loss Risk

*# Error: "Unable to recover state from slurmctld.state"*

*# Emergency Recovery (CAUSES JOB LOSS!)*

```
systemctl stop slurmctld  
cd /var/lib/slurm  
cp slurmctld.state slurmctld.state.backup.$(date +%Y%m%d_%H%M)  
rm -f slurmctld.state  
systemctl start slurmctld
```

# High Availability Failover

## Controller Failover Procedure

```
# Primary controller failed  
# On backup controller:  
systemctl status slurmctld  
grep "PRIMARY\|BACKUP" /etc/slurm/slurm.conf  
  
# Force failover if backup doesn't auto-activate  
scontrol takeover  
  
# Repair and failback:  
# 1. Fix primary controller  
# 2. Update slurm.conf if needed  
# 3. scontrol reconfig on all nodes  
# 4. Let primary resume naturally
```

**Best Practice:** Test failover procedures regularly in maintenance windows



# Database (slurmdbd) Failures

## Database Daemon Issues

*# Symptoms: "SLURM accounting storage is disabled"*

*# Diagnosis*

```
systemctl status slurmdbd
```

```
tail -100 /var/log/slurm/slurmdbd.log
```

```
mysql -u slurm -p -h localhost -e "SHOW DATABASES;"
```

*# Recovery*

```
systemctl start slurmdbd
```

*# Debug mode:*

```
slurmdbd -D -vvv
```

## Database Corruption

*# Error: "Table './slurm\_acct\_db/job\_table' is marked as crashed"*

*# Repair*

```
systemctl stop slurmdbd
```

```
mysql -u slurm -p slurm_acct_db
```

```
CHECK TABLE job_table;
```

```
REPAIR TABLE job_table;
```

*# Mass repair:*

```
mysqlcheck -u slurm -p --repair slurm_acct_db
```

# Database Connection Issues

## Connection Exhaustion

*# Error: "1040 Too many connections"*

### *# Diagnosis*

```
mysql -u root -p -e "SHOW PROCESSLIST;"
```

```
mysql -u root -p -e "SHOW STATUS LIKE 'Threads_connected';"
```

*# Fix in /etc/mysql/mariadb.conf.d/50-server.cnf:*

```
max_connections = 1000
```

```
wait_timeout = 28800
```

*# In slurmdbd.conf:*

```
MaxQueryTimeRange=MONTH # Instead of INFINITE
```

```
systemctl restart mariadb slurmdbd
```

# Compute Node (`slurmd`) Failures

## Node Daemon Recovery

*# Node shows as "down" or "not responding"*

*# Diagnosis*

```
systemctl status slurmd
```

```
journalctl -u slurmd -n 50
```

```
tail -100 /var/log/slurm/slurmd.log
```

*# Recovery*

```
systemctl start slurmd
```

```
scontrol update NodeName=node001 State=resume Reason="slurmd restarted"
```

*# Mass recovery:*

```
pdsh -w node[001-100] "systemctl restart slurmd"
```

```
scontrol update NodeName=node[001-100] State=resume Reason="mass restart"
```

# Hardware Configuration Mismatch

## CPU/Memory Detection Issues

*# Error: "Node configuration differs from hardware: CPUs=64:128(hw)"*

### *# Diagnosis*

```
scontrol show node node001 | grep -E "CPUs|RealMemory|Gres"
```

### *# On compute node:*

```
nproc
```

```
free -g
```

```
nvidia-smi -L
```

### *# Resolution Options:*

*# 1. Update slurm.conf to match hardware*

*# 2. Use hardware detection*

```
slurmd -C # Print actual hardware config
```

**Best Practice:** Use `slurmd -C` output to generate initial node configs

# Authentication (Munge) Failures

## Munge Key Issues - Security Critical

*# Error: "Munge credential decode failed: Invalid credential"*

*# Diagnosis*

`munge -n | unmunge` *# Test locally*

`pdsh -w node[001-100] "munge -n"` *# Test all nodes*

*# Recovery - Sync keys*

`systemctl stop munge`

`pdcp -w node[001-100] /etc/munge/munge.key /etc/munge/`

`pdsh -w node[001-100] "chown munge:munge /etc/munge/munge.key"`

`pdsh -w node[001-100] "chmod 400 /etc/munge/munge.key"`

`pdsh -w node[001-100] "systemctl restart munge"`

## Permission Problems

*# Fix munge directory permissions*

`chown munge:munge /var/lib/munge /var/log/munge /run/munge`

`chmod 755 /var/lib/munge /var/log/munge /run/munge`

# Network/Firewall Issues

## Controller Unreachable

### *# Diagnosis*

```
ping slurmctld-host  
telnet slurmctld-host 6817  
ss -tlnp | grep 6817  
iptables -L -n | grep 6817
```

### *# Resolution - Open required ports*

```
firewall-cmd --permanent --add-port=6817/tcp # slurmctld  
firewall-cmd --permanent --add-port=6818/tcp # slurmd  
firewall-cmd --permanent --add-port=6819/tcp # slurmdbd  
firewall-cmd --reload
```

## Critical Ports:

- 6817: slurmctld (controller)
- 6818: slurmd (compute nodes)
- 6819: slurmdbd (database)

# Critical Recovery – Nuclear Option

## Complete Cluster Recovery Procedure

*# When everything is broken - last resort*

*# 1. Stop all services*

```
pdsh -w node[001-100] "systemctl stop slurmd"  
systemctl stop slurmctld slurmdbd
```

*# 2. Backup critical state*

```
cp -r /var/lib/slurm /backup/slurm_state_$(date +%Y%m%d_%H%M)  
mysqldump -u slurm -p slurm_acct_db > /backup/slurm_db_$(date +%Y%m%d).sql
```

*# 3. Clean start (removes job state!)*

```
rm -f /var/lib/slurm/slurmctld.state*
```

*# 4. Start services in correct order*

```
systemctl start slurmdbd  
systemctl start slurmctld  
pdsh -w node[001-100] "systemctl start slurmd"
```

*# 5. Resume all nodes*

```
scontrol update NodeName=ALL State=resume Reason="cluster restart"
```

# Database Emergency Recovery

## Recreate Corrupted Accounting Database

*# If accounting DB is completely corrupted*

```
systemctl stop slurmdbd slurmctld
```

*# Recreate database*

```
mysql -u root -p
```

```
DROP DATABASE slurm_acct_db;
```

```
CREATE DATABASE slurm_acct_db;
```

```
GRANT ALL ON slurm_acct_db.* TO 'slurm'@'localhost';
```

*# Reinitialize accounting*

```
sacctmgr -i create cluster cluster_name
```

```
sacctmgr -i add account root Cluster=cluster_name
```

```
sacctmgr -i add user root DefaultAccount=root
```

```
systemctl start slurmdbd slurmctld
```



# Configuration Synchronization

## Config File Mismatch Issues

*# Error: "Node appears to have different slurm.conf"*

*# Diagnosis*

```
md5sum /etc/slurm/slurm.conf # Controller
```

```
pdsh -w node[001-100] "md5sum /etc/slurm/slurm.conf" # All nodes
```

*# Resolution*

```
pdcp -w node[001-100] /etc/slurm/slurm.conf /etc/slurm/
```

```
pdsh -w node[001-100] "systemctl reload slurmd"
```

```
scontrol reconfig
```

## Clock Synchronization

*# Error: "Message time stamp is too far in the future"*

*# Check time sync*

```
pdsh -w node[001-100] date
```

```
chrony sources -v
```

*# Force sync*

```
pdsh -w node[001-100] "chrony makestep"
```

# Monitoring & Health Checks

## Automated Health Check Script

```
#!/bin/bash
# Controller health
if ! systemctl is-active slurmctld >/dev/null; then
    echo "CRITICAL: slurmctld is down"
    logger "SLURM: slurmctld service failed"
fi

# Database health
if ! systemctl is-active slurmdbd >/dev/null; then
    echo "CRITICAL: slurmdbd is down"
fi

# Node health
DOWN_NODES=$(sinfo -h -t down -o %N)
[ ! -z "$DOWN_NODES" ] && echo "WARNING: Down nodes: $DOWN_NODES"

# Database connectivity
if ! mysql -u slurm -p[password] -e "SELECT 1;" >/dev/null 2>&1; then
    echo "CRITICAL: Database connection failed"
fi
```

Deploy: Run every 5 minutes via cron, integrate with monitoring system

# Backup Strategy

## Automated Daily Backup

```
#!/bin/bash
DATE=$(date +%Y%m%d)
BACKUP_DIR="/backup/slurm"

# State files backup
mkdir -p $BACKUP_DIR/state
cp -r /var/lib/slurm/* $BACKUP_DIR/state/slurm_state_$DATE/

# Database backup
mysqldump -u slurm -p[password] slurm_acct_db | \
    gzip > $BACKUP_DIR/slurm_acct_db_$DATE.sql.gz

# Configuration backup
mkdir -p $BACKUP_DIR/config
cp /etc/slurm/* $BACKUP_DIR/config/slurm_config_$DATE/

# Cleanup (keep 30 days)
find $BACKUP_DIR -name "*" -mtime +30 -delete
```

# Prevention Best Practices

## Service Start Order

# Always follow correct startup sequence:

1. `slurmdbd` # Database first
2. `slurmctld` # Controller second
3. `slurmd` # Compute nodes last

## Regular Maintenance Tasks

- Weekly: Check for down nodes, review logs
- Monthly: Test backup/restore, failover procedures
- Quarterly: Update documentation, review capacity

## Change Management

# Before any config changes:

1. Backup current state
2. Test `in` development environment
3. Schedule maintenance window
4. Have rollback plan ready
5. Monitor post-change

# Recovery Time Objectives

## Recovery Time Objectives

Failure Type	RTO	Critical Steps
-----	-----	-----
Single node down	2-5 min	`systemctl restart slurmd`
Config mismatch	5-10 min	Sync configs, `scontrol reconfig`
slurmctld crash	1-3 min	`systemctl start slurmctld`
State corruption	15-30 min	Restore from backup
Database corruption	30-60 min	Repair tables or restore DB
Complete cluster failure	2-4 hours	Full recovery procedure

## Escalation Path

1. L1: Basic service restarts, node resume
2. L2: Config sync, log analysis, backup restore
3. L3: Database recovery, network troubleshooting
4. Vendor: Hardware failures, software bugs

# Common Pitfalls & Gotchas

## What NOT to Do

- ✗ **Never** delete state files without backup
- ✗ **Never** restart services without checking dependencies
- ✗ **Never** modify database directly without stopping slurmdbd
- ✗ **Never** ignore authentication (munge) errors

## Emergency Shortcuts That Backfire

*# DON'T do these under pressure:*

```
rm -f /var/lib/slurm/*           # Loses all jobs
systemctl restart slurmctld slurmdbd  # Wrong order
scontrol update NodeName=ALL State=down  # Mass outage
```

## Always Remember

- ✓ Backup before changes
- ✓ Check logs first
- ✓ Follow service dependencies
- ✓ Test in non-production first

# Advanced Troubleshooting Tools

## Log Analysis Commands

### *# Real-time monitoring*

```
journalctl -u slurmctld -f  
tail -f /var/log/slurm/slurmctld.log
```

### *# Pattern matching*

```
grep -E "(ERROR|FATAL|WARNING)" /var/log/slurm/*.log  
zgrep "node001" /var/log/slurm/slurmctld.log*
```

### *# Performance analysis*

```
strace -p $(pgrep slurmctld)  
perf top -p $(pgrep slurmctld)
```

## Database Analysis

### *# Connection monitoring*

```
mysql -u root -p -e "SHOW PROCESSLIST;" | grep slurm
```

### *# Query analysis*

```
mysql -u root -p -e "SHOW STATUS LIKE 'Slow_queries';"
```

### *# Table analysis*

```
mysql -u slurm -p slurm_acct_db -e "SHOW TABLE STATUS;"
```

# Key Takeaways

## Recovery Principles

1. Systematic Diagnosis - logs → config → network → hardware
2. Service Dependencies - respect startup order
3. Backup Strategy - automate daily, test monthly
4. Change Control - test first, rollback ready
5. Monitoring - proactive alerts prevent outages

## Most Critical Skills

- Log analysis - 80% of issues are in logs
- Config management - keep systems synchronized
- Database maintenance - accounting is business critical
- Network troubleshooting - foundation of cluster communication

## Emergency Contacts

- Keep vendor support numbers handy
- Maintain escalation procedures
- Document tribal knowledge



# Q&A & Discussion

## Discussion Topics

- Site-specific experiences with Slurm failures
- Local backup/recovery procedures
- Integration with monitoring systems
- Automation opportunities

## Resources

- Slurm Documentation: <https://slurm.schedmd.com/documentation.html>
- Troubleshooting Guide:  
<https://slurm.schedmd.com/troubleshoot.html>
- Mailing Lists: [slurm-users@lists.schedmd.com](mailto:slurm-users@lists.schedmd.com)

## Contact Information

- Internal escalation procedures
- Vendor support contacts
- Emergency response team

# Bonus

## Emergency Command Cheat Sheet

### *# Service status*

```
systemctl status slurmctld slurmdbd
```

```
sinfo -NeI # Node status
```

```
squeue -u $USER -o "%i %t %r" # Job status
```

### *# Quick recovery*

```
systemctl restart slurmctld
```

```
scontrol update NodeName=X State=resume Reason="fixed"
```

```
scontrol reconfig # Reload config
```

### *# Emergency contacts*

```
tail -100 /var/log/slurm/slurmctld.log
```

```
journalctl -u slurmctld -n 50
```

```
mysql -u slurm -p slurm_acct_db
```

### *# Backup restore*

```
systemctl stop slurmctld
```

```
cp /backup/slurmctld.state.YYYYMMDD /var/lib/slurm/slurmctld.state
```

```
systemctl start slurmctld
```

# MPI/Parallel Execution Failures

## MPI Version Mismatch – #1 User Issue

*# Error: "OPAL ERROR: Unreachable in file pmix3x\_client.c"*

*# Diagnosis*

```
module list
```

```
which mpirun
```

```
ldd $(which mpirun) | grep pmix
```

```
srun --mpi=list
```

*# Resolution*

```
module purge
```

```
module load gcc/12.1.0 openmpi/4.1.5
```

```
srun --mpi=pmix_v4 ./mpi_app
```

## InfiniBand/Network Issues

*# Test connectivity*

```
srun -N 2 hostname
```

```
srun -N 2 --ntasks=2 ib_write_bw
```

*# Force IB transport*

```
export OMPI_MCA_btl=self,vader,openib
```

```
export UCX_NET_DEVICES=mlx5_0:1
```